



*Triakis Corporation*

---

## **Simulator Test Document**

**For the**

# **Shuttle Remote Manipulator System**

**A NASA CI03  
SARP Initiative 583  
IVV-70 Project**



## Table of Contents

1	Introduction .....	4
1.1	Purpose .....	4
1.2	References .....	4
2	Simulator Test Procedure .....	5
2.1	Installation .....	5
2.2	Building the Executable .....	5
2.3	Running .....	5
2.4	Conducting the Test .....	5
2.5	Test Environments .....	5
2.5.1	PC Based Tests.....	5
2.5.2	Evaluation Board Testing.....	6
2.5.3	Customer Driver Card Testing .....	7
2.5.4	List of Programs and Files Used for Testing .....	7
2.6	Running the Customer System MPC555 Simulator .....	7
2.7	Running the MPC555 Simulator .....	7
2.8	Running the Axiom MPC555 Simulator .....	7
2.9	Configuring the Axiom Evaluation Board .....	8
2.10	Running a TestSoftware App on the Axiom Eval Board.....	8
3	Tests.....	8
3.1	MPC555 Instruction Tests .....	8
3.1.1	Simulator Test .....	8
3.1.2	Evaluation Board Test .....	9
3.1.3	Test Analysis.....	9
3.2	Simulator Parts Tests (Except MPC555) .....	9
3.2.1	BankedFROM_eld_a_1 and A28F400_tri_a_1 .....	9
3.2.2	ByteSw_tri_a_1.....	10
3.2.3	CardPowerSupply_eld_a_1 .....	10
3.2.4	CargoDoorSystem_eld_a_1 .....	11
3.2.5	ClockDivider_eld_a_1 .....	11
3.2.6	ComDiscDet_eld_a_1.....	11
3.2.7	Counter_16BitUp_tri_a_1.....	12
3.2.8	Crystal_tri_a_1 .....	12
3.2.9	ExecMark.....	12
3.2.10	FaultIsoCntrlSw_eld_a_1 .....	12
3.2.11	GndDiscDet_eld_a_1 .....	12
3.2.12	InputCond_eld_a_1 .....	13
3.2.13	LandingGearSystem_eld_a_1.....	13
3.2.14	LEDIndicator_tri_a_1 .....	13
3.2.15	LGAnimSystem_eld_a_1 .....	13
3.2.16	LGLeverSystem_eld_a_1.....	13
3.2.17	M39016_tri_a_1 .....	13
3.2.18	MC145050_tri_a_1 .....	13
3.2.19	MC33298_tri_a_1.....	14
3.2.20	MC74HC589_tri_a_1 .....	14
3.2.21	MM74HC594_tri_a_1 .....	14
3.2.22	OutputCond_eld_a_1 .....	15
3.2.23	PCSDecoder_eld_a_1 .....	15
3.2.24	PseudoP5_eld_a_1 (PseudoP5_eld_bSystemcd_1) .....	15
3.2.25	ResetControl_eld_a_1 .....	15
3.2.26	RMSControl_eld_a_1 .....	15
3.2.27	RS422Comm_eld_a_1 .....	16
3.2.28	STK12C68_tri_a_1.....	16
3.3	MPC555 Testing .....	16



3.3.1	MPC555_tria_1 .....	16
3.3.2	MPC555Clock_tria_1 .....	16
3.3.3	MPC555CpuCore_tria_1.....	16
3.3.4	MPC555DEC_tria_1.....	16
3.3.5	MPC555DMPU_tria_1 .....	17
3.3.6	MPC555DPTRAM_tria_1 .....	17
3.3.7	MPC555EBIMemCtrl_tria_1 .....	17
3.3.8	MPC555FlashCtrl_tria_1 .....	17
3.3.9	MPC555GPIO_tria_1 .....	17
3.3.10	MPC555IMPU_tria_1.....	17
3.3.11	MPC555IntCtrl_tria_1 .....	17
3.3.12	MPC555MachineCtrl_tria_1 .....	17
3.3.13	MPC555MIOS1_tria_1.....	17
3.3.14	MPC555PIT_tria_1.....	18
3.3.15	MPC555QADC_tria_1.....	18
3.3.16	MPC555QSMCM_tria_1 .....	18
3.3.17	MPC555RAMCtrl_tria_1 .....	18
3.3.18	MPC555RTC_tria_1 .....	18
3.3.19	MPC555SCI_tria_1 .....	18
3.3.20	MPC555TB_tria_1 .....	18
3.3.21	MPC555TouCAN_tria_1 .....	19
3.3.22	MPC555TPU3_tria_1 .....	19
3.3.23	MPC555UIMB_tria_1 .....	19
3.3.24	MPC555USIU_tria_1 .....	19
3.3.25	MPC555WDT_tria_1 .....	19
3.3.26	PPCCpu_tria_1 .....	19
3.3.27	TPUDIO_tria_1 .....	19
3.3.28	TPUOC_tria_1.....	19
3.3.29	TPUQOM_tria_1 .....	19
3.3.30	TPUSIOP_tria_1 .....	20
3.3.31	TPUUART_tria_1 .....	20
3.4	Report Output Generation Test.....	20
3.4.1	Subroutine Calling Trace Test.....	20
3.4.2	Path Coverage Analysis Test (Jump Markers Test).....	22
3.4.3	Execution Markers Test .....	26
3.5	Instruction Timing Test .....	31

## Table of Figures

Figure 1:	Configuration for testing standard block parts and MPC555 .....	5
Figure 2:	Simulated MPC555 Axiom Evaluation Board Test Configuration.....	6
Figure 3:	SPI bus, PseudoP5, and Other HW Parts Test Configuration .....	6
Figure 4:	Axiom Evaluation Board Comparative Test Configurations.....	6



## 1 Introduction

This specification has been developed to support a research project funded by the NASA Software Assurance Research Program (SARP) during the fiscal year 2003 Center Initiatives (CI03) effort. A system-level, executable specification (ES) based simulation of the Shuttle Remote Manipulator System (SRMS) has been created from the requirements specified in the System Requirements (SARP-I583-001) and Simulator Requirements (SARP-I583-002) Specifications, and used as a vehicle for exploring the concepts described in section 2 of Triakis proposal number TC\_G020614.

For a project slated for hardware development, this document would provide the methods and tests used to verify the SRMS simulator itself. For the purposes of this project, this document is intended to be somewhat illustrative, as it will present the methods & tests used for verifying an actual avionics project MPC555-based simulator. Since the MPC555 is at the core of the SRMS DE simulator, the tests used for verifying the MPC555 part found herein apply equally to the SRMS project simulator.

The original simulator test document has been edited to preserve confidentiality of potential customer-sensitive information. Our apologies for any resulting confusion this may have caused, please contact a project PI if you would like clarification on any details about how we verify a simulator.

### 1.1 Purpose

The SRMS simulator has been developed to facilitate the research goals stated in Triakis proposal number TC\_G020614. System components and functions of the real-world SRMS that are not required to support our research goals have been omitted. Please refer to the System Requirements Specification (SARP-I583-001) for a detailed explanation of the project purpose.

This document presents a test procedure developed for verifying the accuracy and completeness of the simulator. The intent is to meet or exceed the requirements in DO178B for the verification of a test tool.

### 1.2 References

- SARP-I583-001 System Requirements Specification for the Shuttle Remote Manipulator System
- SARP-I583-002 Simulator Requirements Specification for the Shuttle Remote Manipulator System
- SARP-I583-302 System Test Document for the Shuttle Remote Manipulator System
- TC\_G020614 Triakis proposal to NASA for the SARP (Solicitation No: NRA SARP 0201), 14 June 2002
- RTCA DO-178B Software Consideration in Airborne Systems and Equipment Certification
- Triakis Doc TORSWDD xxx-xxxx Tool Operational Requirements Document and Software Design Document, MPC555 Based Simulator
- Triakis Doc MAN xxx-xxxx Users Manual, MPC555 Based Simulator
- Triakis Doc SWTP xxx-xxxx Software Test Procedure, MPC555 Based Simulator
- Triakis Doc SWTR xxx-xxxx Software Test Report, MPC555 Based Simulator
- Triakis Doc VDD xxx-xxxx Version Description Document, MPC555 Based Simulator
- Triakis Software Development Kit Users Manual
- Triakis SDK Reference
- Triakis Part Specific Documents
- MPC555 Microcontroller User Manual, Motorola, Inc.
- Customer Object Oriented TestSoftware Document
- Green Hills Multi Building and Editing Manual
- Green Hills C++ User's Guide
- Green Hills Embedded PowerPC Development Guide
- Green Hills Multi Debugging Manual



## 2 Simulator Test Procedure

All testing described herein is based on the MPC555 Based Simulator Release 1.

### 2.1 Installation

Perform the following steps:

1. Select a directory to install the MPC555 Based Simulator
2. Copy the ESSS.zip file to the directory and unzip.
3. Open Windows Explorer and go to directory ESSS\_Release\_1\TriakisSDK
4. Double click on setup.exe and follow installation instructions.
5. Examine the installed files and verify that they match date/time/size with Release 1 in VDD-xxx-xxx Rev -.

### 2.2 Building the Executable

All software is delivered pre-built.

### 2.3 Running

Follow instructions in sections 2.6 through 2.11 to run specific, selected applications as called for in the tests.

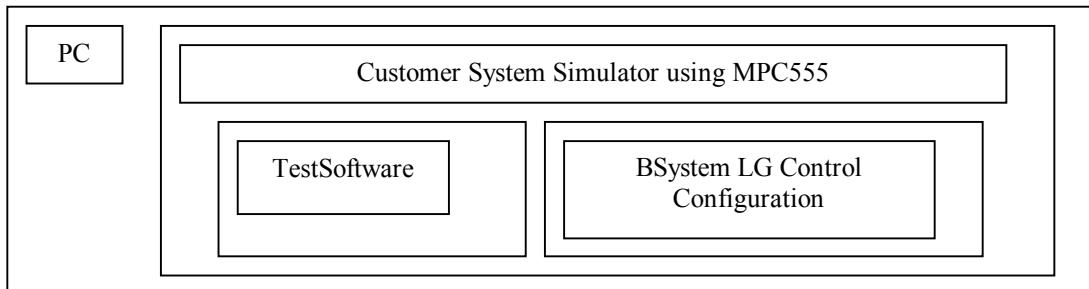
### 2.4 Conducting the Test

The tests have been run by Triakis using release 1 of the software, and maintained under configuration control. The results of each test have been recorded for the test report. Test failure records are preserved and logged as SPRs.

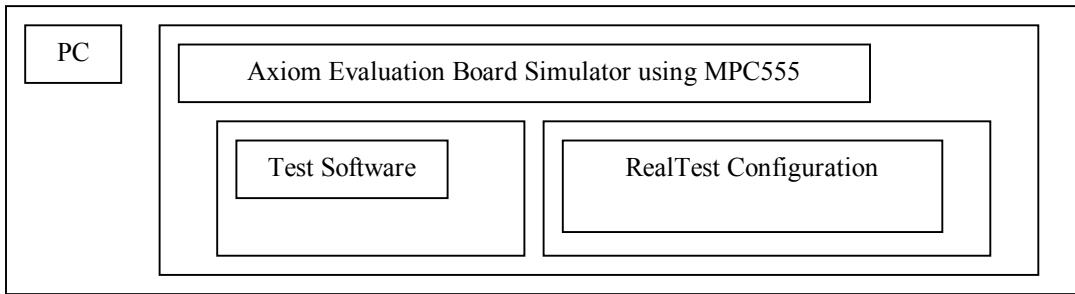
### 2.5 Test Environments

#### 2.5.1 PC Based Tests

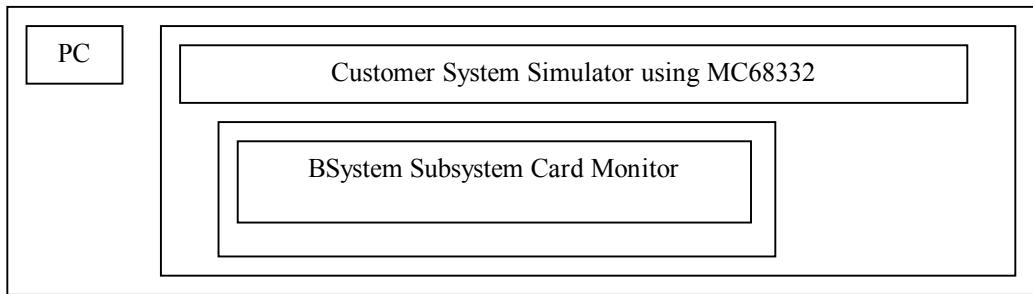
These tests are run completely on the PC. They consist of an auto-test and/or the OO Test Software with a configuration table, or a separate test program such as a monitor (RMS). Figures 1 through 3 depict the various configurations used for testing.



**Figure 1: Configuration for testing standard block parts and MPC555**



**Figure 2: Simulated MPC555 Axiom Evaluation Board Test Configuration**



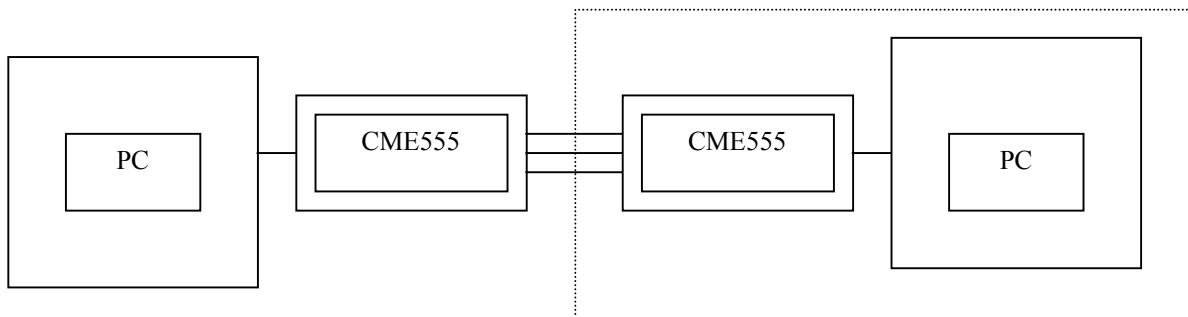
**Figure 3: SPI bus, PseudoP5, and Other HW Parts Test Configuration**

The use of the System PSEU RMS software that has been tested in real hardware is a very powerful tool for the verification of the simulated hardware.

## 2.5.2 Evaluation Board Testing

Various simulated MPC555 on-chip peripherals may be tested with a program that runs on one or two evaluation boards. Figure 4 illustrates the two Axiom evaluation board test configurations with the dashed line surrounding the optional dual-board equipment set-up. All testing performed for this effort used the single board configuration.

The tests are written such that the results may be read out of memory in the evaluation board. These tests show that the simulation matches the real hardware in its behavior. The Axiom Evaluation Board is also referred to as the CME555.



**Figure 4: Axiom Evaluation Board Comparative Test Configurations**



### **2.5.3 Customer Driver Card Testing**

Some peripheral devices external to the MPC555 that are contained in Customer Standard Blocks can be tested by connecting an evaluation board to an Customer driver card.

### **2.5.4 List of Programs and Files Used for Testing**

<b>File Name</b>	<b>Description</b>
AutoMonitor.exe	Program to automatically interface to the CME555 Evaluation Board
CustomerSystem.exe in the CustomerSystemMPC555 Directory	
CustomerSystem.exe in the CustomerSystemMC68332 Directory	
CustSys.exe	
Axiom.exe	
MULTI.EXE	Green Hills Power PC Compiler, IDE, and debugger
msdev.exe	Microsoft Visual C++ 6.0 Compiler and IDE
triserv.exe	Triakis debugger server for the GH Multi
LGControl.bld	TestSoftware program for CustomerSystem.exe (MPC555 version)
C5SPCB.bld	TestSoftware program for CustSys.exe
RealTest.bld	TestSoftware program for Axiom.exe tests
SimInstrTest.txt	Result of instruction tests run in simulator
RealInstrTest.txt	Result of instruction tests run on evaluation board
InstrLog.txt	Result of running the subroutine calling trace test
JumpMark.txt	Result of running the path coverage test.
ExecMark.dat	Result of running the execution markers test.
terminal.exe	Windows dumb terminal program

## **2.6 Running the Customer System MPC555 Simulator**

Open MS Visual C++ and select the ESSSProject\CustomerSystemMPC555\CustomerSystem.dsw project.

## **2.7 Running the MPC555 Simulator**

Open MS Visual C++ and select the ESSSProject\CustSys\CustSys.dsw project.

## **2.8 Running the Axiom MPC555 Simulator**

Open MS Visual C++ and select the ESSSProject\Axiom\Axiom.dsw project.



## 2.9 Configuring the Axiom Evaluation Board

Set the Axiom Evaluation board switches as follows:

RAM_SEL	Jmp 1
FLSH_SEL	Jmp 3
M_SEL	None
MEM_OPT	Jmp 5 and Jmp 7
Mode Sw 1	(starting at sw 1) OFF ON OFF OFF OFF OFF OFF OFF
Mode Sw 2	(starting at sw 1) OFF OFF OFF OFF OFF OFF OFF OFF

## 2.10 Running a TestSoftware App on the Axiom Eval Board

1. Remove power from the development board.
2. Connect the Macraigor Wiggler to your PC's LPT1 port.
3. Connect the Serial Cable from the development board to your PC's COM 1 port.
4. Run the Windows Terminal program (Terminal.exe).
5. In the terminal program, click on Settings/Communications, and set 9600 baud, COM 1.
6. Apply power to the Wiggler.
7. Apply power to the development board.
8. Open Multi for PPC.
9. Click on File/Open Project in Builder.
10. Select the TestSoftware application.
11. Click on Remote/Connect to Target
12. Select the On-Chip Debugger driver: ocdserv lpt1 ppc555 -s evb555.ocd  
Multi should show two windows that indicate the connection has been established to the wiggler.
13. Connect the BDM connector to the development board.
14. On the development board, press the PWR\_ON RESET switch. The Reset LED should flash on briefly.
15. Click on Debug/Debug <name of application>, or click on the "bug" symbol.
16. Recompute the Symbol file if requested.
17. Click on the "Next" button. A progress bar will appear while the target software is loading.
18. Click on the "Run" button.

## 3 Tests

### 3.1 MPC555 Instruction Tests

There are four instruction test sections: Arithmetic, Branch, Floating Point, and Load/Store Instructions. The tests are run in the Axiom board simulator and on the actual Axiom evaluation board for direct comparison of results.

#### 3.1.1 Simulator Test

1. In the Axiom MPC555 Simulator, put the following code in the Auto.cpp file, compile, and run the simulator:

```
Message("Hello World");

World *world = (World *)Sim::ExtNameToPartPtr(      "The World");

PPCCpu_tri_a_1 *cpu = (PPCCpu_tri_a_1 *)Sim::ExtNameToPartPtr(
    "The World/Axiom Test/MPC555 SBC/MPC555 Wrapper/MPC555/Cpu Core/PPC Cpu");
```



```
Terminal *terminal = (Terminal *)Sim:::ExtNameToPartPtr(
    "The World/Axiom Test/Terminal");

LogicSupply_tri_a_1 *logsupply = (LogicSupply_tri_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Axiom Test/Cpu-Logic Supply");

CpuList[0] = (Cpu *)cpu; // For debugger

char sysName[80];
strcpy(sysName, world->GetSystemName());
Message("System Name:");
Message(sysName);

BOOL success = world->GenerateSimFiles(".\\GenSimFiles", TRUE);

world->RestoreWindows("monitor.stu");

// MPC555 Instruction Test
#if 1
cpu->SetSpeedFactor(3);

eq.Delay(1.0);

logsupply->SetPowerState(TRUE);

eq.Delay(1.0);

RunInstructionTest(terminal);
#endif

Message("Test Finished");

for(;;)
    eq.Delay(1.0);
```

2. When the test is finished, save the result file, InstrTest.txt as SimInstrTest.txt.

### **3.1.2 Evaluation Board Test**

1. Ready the Axiom Evaluation Board per section 2.11, steps 1-14, with the RealTest.bld application.
2. Run the AutoMonitor program, in a DOS window type: AutoMonitor RealInstrTest.txt
3. Verify a DOS window appears with the message: Reset Eval Board
4. Finish steps 15-18 in section 2.11.
5. Verify the message: WAIT appears in the DOS window.

### **3.1.3 Test Analysis**

Compare the files: SimInstrTest.txt and RealInstrTest.txt, and verify that they are identical.

## **3.2 Simulator Parts Tests (Except MPC555)**

### **3.2.1 BankedFROM\_eld\_a\_1 and A28F400\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/RMS Control”
3. Open Window “The World/Customer System-300/Customer Pseu/RMS Jumpers/SubSel1”
4. Open Window “The World/Customer System-300/Customer Pseu/RMS Jumpers/SubSel2”
5. Open Window “The World/Customer System-300/Power Busses/Bus 9”
6. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Terminal”



7. The system should initialize in the RMS mode. In the Terminal window, type the monitor command: MDW 400000.
8. Verify the memory contents are FF FF FF FF ...
9. In the Terminal window, type the monitor command MDW 460000.
10. Verify the memory contents are 7C 00 00 A6 ...
11. Set SubSel1 and SubSel2 to TRUE.
12. Toggle Bus power, set Bus 9 to zero volts, then back to 28 volts.
13. Verify the RMS Enable Stat in the RMS Control window is FALSE.
14. In the Terminal window, type the monitor command MDW 400000.
15. Verify the memory contents are 7C 00 00 A6 ...
16. In the Terminal window, type the monitor command MDW 460000.
17. Verify the memory contents are FF FF FF FF ...

### **3.2.2 ByteSw\_tri\_a\_1**

See test 3.3.2.

This part is verified to work if the clock and phase lock loop is set up in the right mode after power up.

### **3.2.3 CardPowerSupply\_eld\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open window "Customer System-300/Power Busses/Bus 9"
3. Verify set to 28.0 volts.
4. Open window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Power Supply".
5. Bring up the Power Supply Trace window and set to trace all signals.
6. Use break control to break on all signals.
7. Stop the Power Supply and step once.
8. Verify:

INPUT\_PWR\_LOWINT is FALSE

LOGICAL\_5VDC is TRUE

PLUS15VDC is 15.0

PLUS15VDC\_MON is 15.0

MINUS15VDC is -15.0

MINUS15VDC\_MON is -15.0

1. Note the time.
2. Set Bus 9 voltage to 0.0.
3. Step the Power Supply until NOTV28LOWINT goes FALSE.
4. Verify the time is >100 ms and <200 ms from the time in step 1.
5. Step the Power Supply until LOGICAL\_5VDC goes FALSE.
6. Verify the time is >5 and <15 ms from the time in step 3.
7. Verify

INPUT\_PWR\_LOWINT is TRUE

LOGICAL\_5VDC is FALSE

PLUS15VDC is 0.0

PLUS15VDC\_MON is 0.0

MINUS15VDC is 0.0

MINUS15VDC\_MON is 0.0

1. Note the time



2. Set Bus 9 voltage to 28.0.
3. Step the Power Supply once.
4. Verify INPUT\_PWR\_LOWINT is FALSE and LOGICAL\_5VDC is FALSE
5. Step the Power Supply until LOGICAL\_5VDC goes TRUE
6. Verify the time is >15 and <25 ms from the time in step 1.
7. Verify:

INPUT\_PWR\_LOWINT is FALSE

LOGICAL\_5VDC is TRUE

PLUS15VDC is 15.0

PLUS15VDC\_MON is 15.0

MINUS15VDC is -15.0

MINUS15VDC\_MON is -15.0

### 3.2.4 CargoDoorSystem\_eld\_a\_1

See PseudoP5 test

### 3.2.5 ClockDivider\_eld\_a\_1

1. Run the CustomerSystemMPC555 Simulator.
2. Open window "Customer System-300/Power Busses/Bus 9"
3. Verify set to 28.0 volts.
4. Open window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/Clock Divider"
5. Bring up the Clock Divider trace window and set all signals (except LOGICAL\_POWER) to be traced.
6. Set Bus 9 voltage to 0 volts.
7. Verify

CLKIN is 0.0 MHz

DIV2OUT is 0.0 MHz

DIV4OUT is 0.0 MHz

DIV8OUT is 0.0 MHz

1. Set Bus 9 voltage to 28.0 volts.
2. Verify

CLKIN is 16.0 MHz

DIV2OUT is 8.0 MHz

DIV4OUT is 4.0 MHz

DIV8OUT is 2.0 MHz

### 3.2.6 ComDiscDet\_eld\_a\_1

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window "The World/Customer System-300/Landing Gear #1/LG Lever"
3. Open Window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Input Conditioner/Gnd Disc Det 1"
4. Verify Gnd Disc Det 1 voltage input and impedance input are both 0.0.



5. Verify the opened input value is FALSE. (The discrete is closed to gnd).
6. Verify vout is 1.4 +- 1.0 volts.
7. Move the Landing Gear #1 LG Lever to the down position.
8. Verify the opened input value is TRUE.
9. Verify vout is 4.0 +- 1.0 volts.

### **3.2.7 Counter\_16BitUp\_tri\_a\_1**

See MPC555MIOS1 test.

### **3.2.8 Crystal\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open window "Customer System-300/Power Busses/Bus 9"
3. Open window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/Crystal"
4. Verify frequency is 16.0 MHz.
5. Set Bus 9 voltage to 0 volts.
6. Verify frequency is 0 MHz.
7. Set Bus 9 voltage to 28 volts.
8. Verify frequency is 16.0 MHz.

### **3.2.9 ExecMark**

See PPCCpu Test.

### **3.2.10 FaultIsoCntrlSw\_eld\_a\_1**

N/A

### **3.2.11 GndDiscDet\_eld\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window "The World/Customer System-300/Landing Gear #1/LG Lever"
3. Open Window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Input Conditioner/Gnd Disc Det 1"
4. Verify Gnd Disc Det 1 voltage input and impedance input are both 0.0.
5. Verify the opened input value is FALSE. (The discrete is closed to gnd).
6. Verify vout is 1.4 +- 1.0 volts.
7. Move the Landing Gear #1 LG Lever to the down position.
8. Verify the opened input value is TRUE.
9. Verify vout is 4.0 +- 1.0 volts.



## 3.2.12 InputCond\_eld\_a\_1

See ComDiscDet\_eld\_a\_1 and GndDisc\_eld\_a\_1 test.

## 3.2.13 LandingGearSystem\_eld\_a\_1

1. Run the CustomerSystemMPC555 Simulator.
2. Click on File/Open Setup. Select lgdisp.stu
3. On the LG Lever, click on the Up button.
4. Verify the LG goes through the sequence: doors open, gear goes up, doors close.

## 3.2.14 LEDIndicator\_tri\_a\_1

N/A

## 3.2.15 LGAnimSystem\_eld\_a\_1

See LandingGearSystem\_eld\_a\_1 test.

## 3.2.16 LGleverSystem\_eld\_a\_1

See LandingGearSystem\_eld\_a\_1 test.

## 3.2.17 M39016\_tri\_a\_1

1. Run the CustomerSystemMC68332 Simulator.
2. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal"
3. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Relay Output/Relay K1"
4. Open the Relay K1 trace window and enable all signals to be traced.
5. At the + prompt on the terminal, execute the DO (driver output command).
6. Type P to move to the second page.
7. Verify driver T2CMD\_1 is OFF.
8. Verify Relay K1 CTRL signal is high resistance > 1000 ohms.
9. Verify Relay K1 NO is in the Relay Opened state.
10. Verify Relay K1 NC is in the Relay Closed state.
11. On the monitor type f, ff, fe, 1, 0.
12. Verify driver T2CMD\_1 is ON.
13. Verify Relay K1 CTRL signal is low resistance < 1 ohms.
14. Verify Relay K1 NO is in the Relay Closed state.
15. Verify Relay K1 NC is in the Relay Opened state.

## 3.2.18 MC145050\_tri\_a\_1

1. Run the CustomerSystemMPC555 Simulator.



2. Open Window "The World/Customer System-300/Cargo Door/Discrete 1"
3. Open Window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Input Conditioner/Gnd Disc Det 3"
4. Open Window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Analog Input/A/D Converter A"
5. Open the A/D Converter A trace window and select the SPI\_OUT signal.
6. Verify Gnd Disc Det 3 voltage input and impedance input are both 0.0.
7. Verify the Gnd Disc Det 3 opened input value is FALSE. (The discrete is closed to gnd).
8. Verify Gnd Disc Det 3 vout is 1.4 +- 1.0 volts.
9. Verify A/D Converter A trace window shows that there are 8 values indicating 0x12a.
10. Push the Discrete 1 Toggle button.
11. Verify the Discrete Opened value indicates TRUE.
12. Verify the Gnd Disc Det 3 opened input value is TRUE.
13. Verify Gnd Disc Det 3 vout is 4.0 +- 1.0 volts.
14. Verify A/D Converter A trace window shows that of the 8 values that were indicating 0x12a, the 3<sup>rd</sup> one has changed to 0x328.

## 3.2.19 MC33298\_tri\_a\_1

1. Run the CustomerSystemMC68332 Simulator.
2. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal"
3. Open window "The World/Customer System-300/Cargo Door/Load 1"
4. At the + prompt on the terminal, execute the DO (driver output command).
5. Verify that Load 1 energized is FALSE.
6. Verify that SNKDRV01 is OFF, and its drain state is OFF.
7. On the monitor type fe, ff, ff, 1, 0.
8. Verify that Load 1 energized is TRUE.
9. Verify that SNKDRV01 is ON, and its drain state is ON.

Note: This part does not implement the analog output model.

## 3.2.20 MC74HC589\_tri\_a\_1

### SPI Bus Parallel Input Latch

1. Run the CustomerSystemMC68332 Simulator.
2. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal"
3. Open window "The World/Customer System-300/Customer Pseu/CD Subsys Select/SubSel0"
4. At the + prompt on the monitor, enter the IL command.
5. Verify SUB\_SEL0 is 0.
6. On the SubSel0 part, toggle state to TRUE.
7. Verify SUB\_SEL0 is 1.

## 3.2.21 MM74HC594\_tri\_a\_1

### SPI Bus Parallel Output Latch

1. Run the CustomerSystemMC68332 Simulator.
2. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal"
3. Open window "The World/Customer System-300/Customer Pseu/Cargo Door Card/Discrete IO/Serial Output Switch"



4. At the + prompt on the terminal, execute the DO (driver output command).
5. Verify that the Serial Output Switch output value is 0xdf.
6. On the monitor type ff, ff, ff, 0, 1.
7. Verify that the Serial Output Switch output value is 0xbff.

## 3.2.22 OutputCond\_eld\_a\_1

N/A

## 3.2.23 PCSDecoder\_eld\_a\_1

This part is tested as a result of testing the SPI Bus devices.

## 3.2.24 PseudoP5\_eld\_a\_1 (PseudoP5\_eld\_bSystemcd\_1)

The PseudoP5 part is designed to be customized for a specific configuration by changing the name of the class, editing arrays in the file, and re-compiling. This test uses the Customer System Cargo Door configuration.

1. Run the CustomerSystemMC68332 Simulator.
2. Open window “The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal”
3. Open window “The World/Customer System-300/Cargo Door/Sensor 1
4. After the + prompt on the monitor, enter the PX command.
5. Verify that there are 12 prox channels displayed.
6. In the Sensor 1 inductance edit window, enter 5.2 and update.
7. Verify the status of SENS1 changes from FAR to NEAR.
8. In the Sensor 1 window, click on the Set Short button.
9. Verify the status of SENS1 changes from NEAR to SHORT.

## 3.2.25 ResetControl\_eld\_a\_1

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Reset Control”
3. Open Window “The World/Customer System-300/Power Busses/Bus 9”
4. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Terminal”
5. Verify WDT Count = 0 in Reset Control.
6. In Reset Control, toggle Addr Err on/off.
7. Verify the monitor re-initializes in the Terminal.
8. Verify WDT Count = 1 in Reset Control.
9. Perform steps 5-8 until WDT Count = 4.
10. Verify further Addr Err toggles have no effect and the monitor does not re-initialize.
11. Toggle Bus power: set Bus 9 to zero volts then back to 28 volts.
12. Verify the monitor in the Terminal re-initializes.
13. Verify WDT Count = 0.

## 3.2.26 RMSControl\_eld\_a\_1

See test 3.2.1.



### **3.2.27 RS422Comm\_eld\_a\_1**

N/A

### **3.2.28 STK12C68\_tri\_a\_1**

1. Run the CustomerSystemMC68332 Simulator.
2. Open window “The World/Customer System-300/Customer Pseu/Cargo Door Card/Terminal”
3. After the + prompt on the monitor, enter the NV command.
4. Answer Y to start the test.
5. Verify “Test Passed”

## **3.3 MPC555 Testing**

Many of the tests above involve the proper operation of the MPC555, and many subcomponents, in a number of applications. The following tests are supplemental and some involve comparing the part with real hardware.

### **3.3.1 MPC555\_tri\_a\_1**

Tested as a result of testing all MPC555 subcomponents.

### **3.3.2 MPC555Clock\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/USIU/Clock/PLL/Reset”
3. Verify CLKOUT is 40MHz
4. Verify PIT Timer is 1MHz
5. Verify TB Timer is 1MHz

### **3.3.3 MPC555CpuCore\_tri\_a\_1**

Tested as a result of testing all CPU Core subcomponents.

### **3.3.4 MPC555DEC\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/USIU/Decrementer”
3. Verify the counter is continuously counting down.



### **3.3.5 MPC555DMPU\_tri\_a\_1**

N/A

### **3.3.6 MPC555DPTRAM\_tri\_a\_1**

N/A

### **3.3.7 MPC555EBIMemCtrl\_tri\_a\_1**

N/A

### **3.3.8 MPC555FlashCtrl\_tri\_a\_1**

N/A

### **3.3.9 MPC555GPIO\_tri\_a\_1**

N/A

### **3.3.10 MPC555IMPU\_tri\_a\_1**

N/A

### **3.3.11 MPC555IntCtrl\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/USIU/Interrupt Ctrl”
3. Open the trace window and select IRQLEVEL signal.
4. Verify that every time the data reads 0x40, the time between events is 8.2+-0.1 ms.

### **3.3.12 MPC555MachineCtrl\_tri\_a\_1**

N/A

### **3.3.13 MPC555MIOS1\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator
2. Open Window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/MIOS1”



3. Click on File/Open Setup, and select timing.stu.
4. The oscilloscope signals are sent by the MIOS1 parallel port. Verify 4 changing logic signals.

### **3.3.14 MPC555PIT\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Open Window "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/USIU/Prog Interrupt Tmr"
3. Verify the PITR register is counting.
4. Open up the trace window and select PITINT.
5. Verify that the interval between interrupts is 8.2+-1 milliseconds.

### **3.3.15 MPC555QADC\_tri\_a\_1**

1. Run the Axiom Simulator.
2. The simulator should come up with the oscilloscope display.
3. Verify that the rjurr[0] and rjurr[4] registers track the voltage from the signal generator.
4. Verify many well spaced interrupts, approximately 2 every 0.5 milliseconds.

### **3.3.16 MPC555QSMCM\_tri\_a\_1**

1. Run the CustomerSystemMC68332 Simulator.
2. The simulator should come up with QSMCM window open.
3. At the monitor prompt, enter the AD command.
4. Verify reasonable values for analog values shown in the monitor window.

### **3.3.17 MPC555RAMCtrl\_tri\_a\_1**

N/A

### **3.3.18 MPC555RTC\_tri\_a\_1**

N/A

### **3.3.19 MPC555SCI\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator.
2. Verify the monitor works. (It uses the SCI for all I/O characters).

### **3.3.20 MPC555TB\_tri\_a\_1**

1. Run the CustomerSystemMPC555 Simulator



2. Open window “ The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/USIU/Time Base”
3. Verify the TBL register is continuously counting up.

### **3.3.21 MPC555TouCAN\_tri\_a\_1**

N/A

### **3.3.22 MPC555TPU3\_tri\_a\_1**

See test 3.3.30.

### **3.3.23 MPC555UIMB\_tri\_a\_1**

See test 3.3.15.

This component handles interrupts from peripherals. This has been verified with the QADC test.

### **3.3.24 MPC555USIU\_tri\_a\_1**

Tested as a result of testing all USIU subcomponents.

### **3.3.25 MPC555WDT\_tri\_a\_1**

N/A

### **3.3.26 PPCCpu\_tri\_a\_1**

See section 3.1, Instruction Tests.

### **3.3.27 TPUDIO\_tri\_a\_1**

N/A

### **3.3.28 TPUOC\_tri\_a\_1**

N/A

### **3.3.29 TPUQOM\_tri\_a\_1**

N/A



### 3.3.30 TPUSIOP\_tri\_a\_1

1. Run the CustomerSystemMPC555 Simulator.
2. Open window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller Core/MPC555 Wrapper/MPC555/TPU3 B/Siop Chan 0-2”
3. Open window “The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Driver Output/Serial Switch 1”
4. Open window “The World/Customer System-300/Landing Gear #1/LG Lever”
5. Raise LG Lever.
6. Verify SPI inputs and outputs from the Siop Chan 0-2 and the Serial Switch 1 track each other.

### 3.3.31 TPUUART\_tri\_a\_1

N/A

## 3.4 Report Output Generation Test

### 3.4.1 Subroutine Calling Trace Test

In the Auto.cpp file, add and execute the following code:

```
PPCCpu_tri_a_1 *cpu_lg1 = (PPCCpu_tri_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller
Core/MPC555 Wrapper/MPC555/Cpu Core/PPC Cpu");

eq.Delay(0.1);

PowerBus_eld_a_1 *bus9 = (PowerBus_eld_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Power Busses/Bus 9");

bus9->SetVoltage(28.0);

// careful, this can generate a huge file
cpu_lg1->SubroutineRecord(TRUE);
eq.Delay(0.125);
cpu_lg1->SubroutineRecord(FALSE);
```

Verify that the file InstrLog.txt is created with the following contents:

```
0.130130 10014 b1
0.130130 22330 bclr
0.130141 214dc b1 -> 00021fdc [_savegpr_27_1]
0.130144 21ff8 bclr
0.130146 214f8 b1 -> 00021348 [_ghs_init_mem]
0.130147 21354 b1 -> 00021fdc [_savegpr_27_1]
0.130150 21ff8 bclr
0.130159 213cc b1 -> 00021bb4 [_ghs_eofn_memcpy]
0.130164 21c60 bclr
0.130167 213cc b1 -> 00021bb4 [_ghs_eofn_memcpy]
0.131886 21c60 bclr
0.131891 2140c b1 -> 00021b48 [memcpy]
0.139904 21bb0 bclr
0.139949 22054 bclr
0.139949 214fc b1 -> 000218ac [_enter]
0.139951 218b8 b1 -> 00021fe4 [_savegpr_29_1]
0.139952 21ff8 bclr
```



```
0.139954 218c8    bl  -> 000217f8 [__ghs_syscall]
0.139955 21808    bl
0.139955 22330    bclr
0.139957 21824    bclr
0.139960 22054 bclr
0.139961 21510 bl  -> 00021e4c [__gh_lock_init]
0.139963 21e58    bl  -> 00021fe4 [_savegpr_29_1]
0.139964 21ff8    bclr
0.139967 22054 bclr
0.139969 21524 bl  -> 00020a9c [iob_init]
0.139970 20aa8    bl  -> 00021fe4 [_savegpr_29_1]
0.139972 21ff8    bclr
0.139976 20aec    bl  -> 000208f4 [flockcreate]
0.139978 20900    bl  -> 00021fe4 [_savegpr_29_1]
0.139979 21ff8    bclr
0.139983 20930    bl  -> 00021ed0 [__ghs_flock_create]
0.139984 21edc    bl  -> 00021fe4 [_savegpr_29_1]
0.139986 21ff8    bclr
0.139989 22054 bclr
0.139991 22054 bclr
0.139992 20af4    bl  -> 000208f4 [flockcreate]
0.139993 20900    bl  -> 00021fe4 [_savegpr_29_1]
0.139995 21ff8    bclr
0.139999 20930    bl  -> 00021ed0 [__ghs_flock_create]
0.140000 21edc    bl  -> 00021fe4 [_savegpr_29_1]
0.140002 21ff8    bclr
0.140005 22054 bclr
0.140007 22054 bclr
0.140008 20afc    bl  -> 000208f4 [flockcreate]
0.140009 20900    bl  -> 00021fe4 [_savegpr_29_1]
0.140011 21ff8    bclr
0.140014 20930    bl  -> 00021ed0 [__ghs_flock_create]
0.140015 21edc    bl  -> 00021fe4 [_savegpr_29_1]
0.140017 21ff8    bclr
0.140020 22054 bclr
0.140023 22054 bclr
0.140026 22054 bclr
0.140029 21550 bl  -> 00021c64 [__gh_signal_init]
0.140030 21c70    bl  -> 00021fe4 [_savegpr_29_1]
0.140032 21ff8    bclr
0.140033 21c80    bl  -> 00021dc8 [GetThreadLocalStorage]
0.140034 21dd4    bl  -> 00021fe4 [_savegpr_29_1]
0.140036 21ff8    bclr
0.140039 22054 bclr
0.140120 22054 bclr
0.140128 215c0 bl  -> 00015064 [ingSystemMPC555.5CLgControlApp..3C03BB39..0]
0.140129 15070    bl  -> 0001e7a8 [_main]
0.140130 1e7b4    bl  -> 00021fe4 [_savegpr_29_1]
0.140132 21ff8    bclr
0.140135 1e7e4    bl  -> 0001e9c4 [atexit]
0.140137 1e9d0    bl  -> 00021fe0 [_savegpr_28_1]
0.140138 21ff8    bclr
0.140140 1e9e0    bl  -> 00021de8 [__ghsLock]
0.140141 21df4    bl  -> 00021fe4 [_savegpr_29_1]
0.140143 21ff8    bclr
0.140146 22054 bclr
0.140148 1ea04    bl  -> 00021970 [__ghs_at_exit]
0.140149 2197c    bl  -> 00021fe4 [_savegpr_29_1]
0.140151 21ff8    bclr
0.140152 21984    bl  -> 00021de8 [__ghsLock]
0.140153 21df4    bl  -> 00021fe4 [_savegpr_29_1]
0.140155 21ff8    bclr
0.140158 22054 bclr
0.140159 2199c    bl  -> 00021e00 [__ghsUnlock]
0.140161 21e0c    bl  -> 00021fe4 [_savegpr_29_1]
0.140162 21ff8    bclr
0.140165 22054 bclr
0.140168 22054 bclr
0.140171 1ea30    bl  -> 00021e00 [__ghsUnlock]
```



```

0.140172 21e0c      bl  -> 00021fe4 [_savegpr_29_1]
0.140174 21ff8      bclr
0.140177 22054      bclr
0.140180 22054      bclr
0.140185 1e808      bcctrl -> 000168c8 [_sti_FwTiming_cpp_tm]
0.140186 168dc      bl  -> 000167ec [ingSystemMPC555.5CLgControlApp..3C03BB3D..0]
0.140194 16858      bclr
0.140195 168ec      bclr
0.140200 22054      bclr
0.140200 15074      bl  -> 000150ec [FwMain_Fv]
0.140201 150f8      bl  -> 00021fa8 [_savegpr_14_1]
0.140207 21ff8      bclr
0.140208 15100      bl  -> 00011d18 [_ct_13ConfigControlFv]
0.140211 11d50      bclr
0.140212 15108      bl  -> 000106dc [_ct_17DataConnectionRdrFv]
. . .(and so on)

```

### 3.4.2 Path Coverage Analysis Test (Jump Markers Test)

In the Auto.cpp file, add and execute the following code:

```

PPCCpu_tri_a_1 *cpu_lg1 = (PPCCpu_tri_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller
Core/MPC555 Wrapper/MPC555/Cpu Core/PPC Cpu");

eq.Delay(0.1);

PowerBus_eld_a_1 *bus9 = (PowerBus_eld_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Power Busses/Bus 9");

bus9->SetVoltage(28.0);

eq.Delay(0.5);

DumpData(lglterm);

cpu_lg1->GetPtrToJumpMarkers()->Clear();
eq.Delay(0.1);
cpu_lg1->GetPtrToJumpMarkers()->Save ("JumpMarks.txt");

```

Verify that the file JumpMarks.txt is created with the following contents:

Address	Jump	No Jump	Symbol	Offset
00002074	x	x		
00002084	x			
000020ac	x			
000103c0	x	x	PeriodicFunc_6SensorFv	+ 24 hex
000103dc	x		555.5CLgControlApp.5Cobj.5CFwSensor.1.19	+ 18 hex
00010420	x	x	555.5CLgControlApp.5Cobj.5CFwSensor.1.21	+ 40 hex
00010438	x	x	555.5CLgControlApp.5Cobj.5CFwSensor.1.27	+ 14 hex
00010470	x		555.5CLgControlApp.5Cobj.5CFwSensor.1.30	+ 34 hex
000106c8	x		ransferData_14DataConnectionFP8DataType	+ 3c hex
000106d8	x		ransferData_14DataConnectionFP8DataType	+ 4c hex
00010914	x		InData_10DataObjectFiP8DataType	+ 4c hex
0001093c	x		55.5CLgControlApp.5Cobj.5CFwDataObj.0.12	+ 24 hex
00010988	x	x	OutData_10DataObjectFiP8DataType	+ 28 hex
00010ac8	x	x	on_10DataObjectFQ2_10DataObject6ToFromi	+ 20 hex
00010ae4	x	x	55.5CLgControlApp.5Cobj.5CFwDataObj.0.33	+ 18 hex
00010b0c	x	x	55.5CLgControlApp.5Cobj.5CFwDataObj.0.34	+ 24 hex
00010b14	x		5.5CLgControlApp.5Cobj.5CFwDataObj.0.35	+ 4 hex
00010b38	x		ingSystemMPC555.5CLgControlApp..3C03BB38..0	+ 10 hex
00010b64	x		55.5CLgControlApp.5Cobj.5CFwDataType.0.1	+ c hex



00010b88	x	_ct_8CharDataFv + 18 hex
00010be8	x	CopyData_8CharDataFP8DataType + 10 hex
00010c04	x	_ct_8BoolDataFv + 18 hex
00010c54	x	CopyData_8BoolDataFP8DataType + 8 hex
00010cf8	x	CopyData_10SensorDataFP8DataType + 20 hex
00010e0c	x	_ct_7IntDataFv + 18 hex
00010e5c	x	CopyData_7IntDataFP8DataType + 8 hex
00011448	x	PeriodicFunc_4QADCFv + 8 hex
00011488	x	PC555.5CLgControlApp.5Cobj.5CFwQADC.1.15 + 3c hex
000114b0	x	PC555.5CLgControlApp.5Cobj.5CFwQADC.1.17 + 24 hex
000114cc	x	PC555.5CLgControlApp.5Cobj.5CFwQADC.1.21 + 18 hex
00011530	x	Queue1_ISR_4QADCFv + 60 hex
00012cdc	x	ClearWatchdog + 20 hex
00012cfc	x	SerialReady + 1c hex
00012d24	x	MPC555.5CLgControlApp.5Cobj.5Cmain.0.140 + 14 hex
00012d9c	x	GetSByte + 1c hex
000150c4	x	ShortPeriodicInt + 1c hex
000150e8	x	HandleISR + 1c hex
00015b70	x	ingSystemMPC555.5CLgControlApp..3C03BB3B..0 + 2c hex
00015b7c	x	55.5CLgControlApp.5Cobj.5CFwRealTime.0.2 + 8 hex
00015cd8	x	lTimeObjectIterator_14RealTimeObjectsSFv + 10 hex
00015cf0	x	tNextRealTimeObject_14RealTimeObjectsSFv + 14 hex
00015d0c	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.29 + 18 hex
00015df0	x	PeriodicTaskIterator_14RealTimeObjectFv + 8 hex
00015e00	x	GetNextSPTask_14RealTimeObjectFv + c hex
00015e14	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.55 + 10 hex
00015e4c	x	_14RealTimeObjectFM14RealTimeObjectFv_v + 34 hex
00015e80	x	_14RealTimeObjectFM14RealTimeObjectFv_v + 68 hex
00015e90	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.59 + c hex
00016038	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.78 + 60 hex
00016074	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.89 + 10 hex
00016084	x	5.5CLgControlApp.5Cobj.5CFwRealTime.0.90 + c hex
00016120	x	HandleISR_15RealTimeControlsSFv + 30 hex
00016134	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.106 + 10 hex
0001614c	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.107 + 14 hex
00016168	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.109 + 18 hex
000161a0	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.110 + 34 hex
000161dc	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.116 + 38 hex
00016230	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.116 + 8c hex
00016258	x	.5CLgControlApp.5Cobj.5CFwRealTime.0.118 + 24 hex
0001625c	x	UnrecognizedISR_15RealTimeControlsSFv + 0 hex
000163d8	x	ingSystemMPC555.5CLgControlApp..3C03BB3C..0 + 0 hex
000165d8	x	PeriodicFunc_11DriverGroupFv + 34 hex
00016618	x	5.5CLgControlApp.5Cobj.5CFwDrvGroup.1.11 + 3c hex
00016658	x	5.5CLgControlApp.5Cobj.5CFwDrvGroup.1.15 + 3c hex
0001667c	x	5.5CLgControlApp.5Cobj.5CFwDrvGroup.1.18 + 20 hex
000166bc	x	5.5CLgControlApp.5Cobj.5CFwDrvGroup.1.20 + 3c hex
00016714	x	5.5CLgControlApp.5Cobj.5CFwDrvGroup.1.24 + 54 hex
00016734	x	GetDriverCommand_11DriverGroupFi + 14 hex
00016750	x	PutDriverStatus_11DriverGroupFiUs + 18 hex
0001687c	x	SetBit_13TimingMonitorFi + 20 hex
000168a4	x	ResetBit_13TimingMonitorFi + 24 hex
00017288	x	PeriodicFunc_7TPUSIOPFv + 34 hex
000172c4	x	55.5CLgControlApp.5Cobj.5CFwTPUSIOP.1.15 + 38 hex
00017318	x	55.5CLgControlApp.5Cobj.5CFwTPUSIOP.1.19 + 50 hex
00017344	x	55.5CLgControlApp.5Cobj.5CFwTPUSIOP.1.26 + 28 hex
00017ef4	x	PeriodicFunc_6DriverFv + 24 hex
00017f3c	x	555.5CLgControlApp.5Cobj.5CFwDriver.1.21 + 44 hex
00017f44	x	555.5CLgControlApp.5Cobj.5CFwDriver.1.26 + 4 hex
00017f88	x	555.5CLgControlApp.5Cobj.5CFwDriver.1.27 + 40 hex
00017f9c	x	555.5CLgControlApp.5Cobj.5CFwDriver.1.27 + 54 hex
00018214	x	PeriodicFunc_14P5SensorStatusFv + 24 hex
00018258	x	5.5CLgControlApp.5Cobj.5CFwP5SenStat.1.9 + 40 hex
00018260	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.14 + 4 hex
00018268	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.14 + c hex
000182a8	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.15 + 3c hex
000182b4	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.21 + 8 hex
000182d8	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.22 + 20 hex
00018314	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.25 + 38 hex



00018388	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.28 + 70 hex
000183c4	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.28 + ac hex
000183e0	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.28 + c8 hex
00018404	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.35 + 20 hex
00018448	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.38 + 40 hex
000184c0	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.41 + 74 hex
000184fc	x	.5CLgControlApp.5Cobj.5CFwP5SenStat.1.41 + b0 hex
00018744	x	PeriodicFunc_8QSMCMSCIFv + 3c hex
00018778	x	5.5CLgControlApp.5Cobj.5CFwQSMCMSCI.1.12 + 30 hex
00019344	x	PeriodicFunc_5QSMCMFv + 0 hex
00019830	x	PeriodicFunc_8DiscreteFv + 2c hex
0001984c	x	5.5CLgControlApp.5Cobj.5CFwDiscrete.1.21 + 18 hex
00019874	x	5.5CLgControlApp.5Cobj.5CFwDiscrete.1.23 + 24 hex
000198a8	x	5.5CLgControlApp.5Cobj.5CFwDiscrete.1.26 + 30 hex
00019a80	x	SaveData_8IioEntryFUs + 2c hex
00019ad4	x	GetData_8IioEntryFPUs + 38 hex
00019e4c	x	PeriodicFunc_12StateMachineFv + 38 hex
00019e68	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.17 + 18 hex
00019ec4	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.19 + 58 hex
00019edc	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.24 + 14 hex
00019f30	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.26 + 50 hex
00019f84	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.33 + 2c hex
00019fa4	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.36 + 1c hex
00019fd0	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.39 + 28 hex
00019ff8	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.42 + 24 hex
0001a024	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.46 + 28 hex
0001a038	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.46 + 3c hex
0001a07c	x	ParseTransString_12StateMachineFPcPin22 + 34 hex
0001a088	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.54 + 8 hex
0001a0d0	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.55 + 24 hex
0001a0dc	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.55 + 30 hex
0001a110	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.59 + 30 hex
0001a120	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.59 + 40 hex
0001a13c	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.64 + 18 hex
0001a148	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.64 + 24 hex
0001a17c	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.67 + 30 hex
0001a18c	x	.5CLgControlApp.5Cobj.5CFwStateMach.1.67 + 40 hex
0001b330	x	PeriodicFunc_8RecorderFv + 1c hex
0001b3c0	x	555.5CLgControlApp.5Cobj.5CFwRecord.1.12 + 8c hex
0001bb30	x	PeriodicFunc_12QSMCMQSPIIIOfv + 80 hex
0001bbe0	x	LgControlApp.5Cobj.5CFwQSMCMQSPIIIIO.2.19 + ac hex
0001bc0c	x	LgControlApp.5Cobj.5CFwQSMCMQSPIIIIO.2.32 + 28 hex
0001bc20	x	LgControlApp.5Cobj.5CFwQSMCMQSPIIIIO.2.34 + 10 hex
0001bc64	x	LgControlApp.5Cobj.5CFwQSMCMQSPIIIIO.2.35 + 40 hex
0001e054	x	PeriodicFunc_8MC145050Fv + 94 hex
0001e068	x	PeriodicFunc_8MC145050Fv + a8 hex
00021ff8	x	_savegpr_31_1 + c hex
00022054	x	_restgpr_31_1 + 10 hex
00042074	x	__gh_va_arg + 1fe58 hex
00042084	x	__gh_va_arg + 1fe68 hex
000420ac	x	__gh_va_arg + 1fe90 hex
000503c0	x	__gh_va_arg + 2e1a4 hex
000503dc	x	__gh_va_arg + 2elc0 hex
00050420	x	__gh_va_arg + 2e204 hex
00050438	x	__gh_va_arg + 2e21c hex
00050470	x	__gh_va_arg + 2e254 hex
000506c8	x	__gh_va_arg + 2e4ac hex
000506d8	x	__gh_va_arg + 2e4bc hex
00050914	x	__gh_va_arg + 2e6f8 hex
0005093c	x	__gh_va_arg + 2e720 hex
00050988	x	__gh_va_arg + 2e76c hex
00050ac8	x	__gh_va_arg + 2e8ac hex
00050ae4	x	__gh_va_arg + 2e8c8 hex
00050b0c	x	__gh_va_arg + 2e8f0 hex
00050b14	x	__gh_va_arg + 2e8f8 hex
00050b38	x	__gh_va_arg + 2e91c hex
00050b64	x	__gh_va_arg + 2e948 hex
00050b88	x	__gh_va_arg + 2e96c hex
00050be8	x	__gh_va_arg + 2e9cc hex



00050c04	x	__gh_va_arg + 2e9e8 hex
00050c54	x	__gh_va_arg + 2ea38 hex
00050cf8	x	__gh_va_arg + 2eadc hex
00050e0c	x	__gh_va_arg + 2ebf0 hex
00050e5c	x	__gh_va_arg + 2ec40 hex
00051448	x	x __gh_va_arg + 2f22c hex
00051488	x	x __gh_va_arg + 2f26c hex
000514b0	x	__gh_va_arg + 2f294 hex
000514cc	x	__gh_va_arg + 2f2b0 hex
00051530	x	x __gh_va_arg + 2f314 hex
00052cdc	x	__gh_va_arg + 30ac0 hex
00052cfc	x	__gh_va_arg + 30ae0 hex
00052d24	x	__gh_va_arg + 30b08 hex
00052d9c	x	__gh_va_arg + 30b80 hex
000550c4	x	__gh_va_arg + 32ea8 hex
000550e8	x	__gh_va_arg + 32ecc hex
00055b70	x	x __gh_va_arg + 33954 hex
00055b7c	x	__gh_va_arg + 33960 hex
00055cd8	x	__gh_va_arg + 33abc hex
00055cf0	x	x __gh_va_arg + 33ad4 hex
00055d0c	x	__gh_va_arg + 33af0 hex
00055df0	x	__gh_va_arg + 33bd4 hex
00055e00	x	x __gh_va_arg + 33be4 hex
00055e14	x	__gh_va_arg + 33bf8 hex
00055e4c	x	__gh_va_arg + 33c30 hex
00055e80	x	__gh_va_arg + 33c64 hex
00055e90	x	__gh_va_arg + 33c74 hex
00056038	x	x __gh_va_arg + 33e1c hex
00056074	x	x __gh_va_arg + 33e58 hex
00056084	x	x __gh_va_arg + 33e68 hex
00056120	x	x __gh_va_arg + 33f04 hex
00056134	x	x __gh_va_arg + 33f18 hex
0005614c	x	x __gh_va_arg + 33f30 hex
00056168	x	__gh_va_arg + 33f4c hex
000561a0	x	x __gh_va_arg + 33f84 hex
000561dc	x	__gh_va_arg + 33fc0 hex
00056230	x	__gh_va_arg + 34014 hex
00056258	x	__gh_va_arg + 3403c hex
0005625c	x	__gh_va_arg + 34040 hex
000563d8	x	__gh_va_arg + 341bc hex
000565d8	x	x __gh_va_arg + 343bc hex
00056618	x	__gh_va_arg + 343fc hex
00056658	x	x __gh_va_arg + 3443c hex
0005667c	x	x __gh_va_arg + 34460 hex
000566bc	x	__gh_va_arg + 344a0 hex
00056714	x	x __gh_va_arg + 344f8 hex
00056734	x	__gh_va_arg + 34518 hex
00056750	x	__gh_va_arg + 34534 hex
0005687c	x	__gh_va_arg + 34660 hex
000568a4	x	__gh_va_arg + 34688 hex
00057288	x	x __gh_va_arg + 3506c hex
000572c4	x	x __gh_va_arg + 350a8 hex
00057318	x	__gh_va_arg + 350fc hex
00057344	x	x __gh_va_arg + 35128 hex
00057ef4	x	x __gh_va_arg + 35cd8 hex
00057f3c	x	__gh_va_arg + 35d20 hex
00057f44	x	__gh_va_arg + 35d28 hex
00057f88	x	__gh_va_arg + 35d6c hex
00057f9c	x	__gh_va_arg + 35d80 hex
00058214	x	x __gh_va_arg + 35ff8 hex
00058258	x	x __gh_va_arg + 3603c hex
00058260	x	__gh_va_arg + 36044 hex
00058268	x	__gh_va_arg + 3604c hex
000582a8	x	x __gh_va_arg + 3608c hex
000582b4	x	__gh_va_arg + 36098 hex
000582d8	x	__gh_va_arg + 360bc hex
00058314	x	__gh_va_arg + 360f8 hex
00058388	x	__gh_va_arg + 3616c hex
000583c4	x	__gh_va_arg + 361a8 hex



```
000583e0    x    x    __gh_va_arg + 361c4 hex
00058404    x    x    __gh_va_arg + 361e8 hex
00058448    x    x    __gh_va_arg + 3622c hex
000584c0    x    x    __gh_va_arg + 362a4 hex
000584fc    x    x    __gh_va_arg + 362e0 hex
00058744    x    x    __gh_va_arg + 36528 hex
00058778    x    x    __gh_va_arg + 3655c hex
00059344    x    x    __gh_va_arg + 37128 hex
00059830    x    x    __gh_va_arg + 37614 hex
0005984c    x    x    __gh_va_arg + 37630 hex
00059874    x    x    __gh_va_arg + 37658 hex
... (and so on)
```

### 3.4.3 Execution Markers Test

In the Auto.cpp file, add and execute the following code:

```
PPCCpu_tri_a_1 *cpu_lg1 = (PPCCpu_tri_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Customer Pseu/Landing Gear #1 Card/Microcontroller
Core/MPC555 Wrapper/MPC555/Cpu Core/PPC Cpu");

eq.Delay(0.1);

PowerBus_eld_a_1 *bus9 = (PowerBus_eld_a_1 *)Sim:::ExtNameToPartPtr(
    "The World/Customer System-300/Power Busses/Bus 9");

bus9->SetVoltage(28.0);

eq.Delay(0.5);

DumpData(lglterm);

eq.Delay(0.1);

ExecutionMarkers *em;
unsigned long start, end;
int pcc;
char buff[80];

cpu_lg1->SetExecutionMarkersEnable(TRUE);
start = cpu_lg1->GetSymbolAddr("PeriodicFunc__4QADCFv");
end = cpu_lg1->GetSymbolAddr("Queue2_ISR__4QADCFv");
em = cpu_lg1->GetPtrToExecutionMarkers();
em->CreateMarkers(start, end);
eq.Delay(0.01);
pcc = em->GetPercentCoverage();
sprintf(buff, "percent coverage: %d", pcc);
Message(buff);
pcc = em->GetLocationCount(start);
sprintf(buff, "loc count: %d", pcc);
Message(buff);
em->Save("ExecMark.dat");
em->DestroyMarkers(); // Do this before creating markers in a different range
```

Verify that the file ExecMarks.dat is created with the following contents:

Execution Report

11440	2
11441	2
11442	2
11443	2
11444	2
11445	2



11446	2
11447	2
11448	2
11449	2
1144a	2
1144b	2
1144c	1
1144d	1
1144e	1
1144f	1
11450	1
11451	1
11452	1
11453	1
11454	64
11455	64
11456	64
11457	64
11458	64
11459	64
1145a	64
1145b	64
1145c	64
1145d	64
1145e	64
1145f	64
11460	64
11461	64
11462	64
11463	64
11464	64
11465	64
11466	64
11467	64
11468	64
11469	64
1146a	64
1146b	64
1146c	64
1146d	64
1146e	64
1146f	64
11470	64
11471	64
11472	64
11473	64
11474	64
11475	64
11476	64
11477	64
11478	64
11479	64
1147a	64
1147b	64
1147c	64
1147d	64
1147e	64
1147f	64
11480	64
11481	64
11482	64
11483	64
11484	65
11485	65
11486	65
11487	65
11488	65
11489	65
1148a	65



1148b	65
1148c	1
1148d	1
1148e	1
1148f	1
11490	1
11491	1
11492	1
11493	1
11494	1
11495	1
11496	1
11497	1
11498	1
11499	1
1149a	1
1149b	1
1149c	1
1149d	1
1149e	1
1149f	1
114a0	1
114a1	1
114a2	1
114a3	1
114a4	1
114a5	1
114a6	1
114a7	1
114a8	1
114a9	1
114aa	1
114ab	1
114ac	1
114ad	1
114ae	1
114af	1
114b0	1
114b1	1
114b2	1
114b3	1
114b4	
114b5	
114b6	
114b7	
114b8	
114b9	
114ba	
114bb	
114bc	
114bd	
114be	
114bf	
114c0	
114c1	
114c2	
114c3	
114c4	
114c5	
114c6	
114c7	
114c8	
114c9	
114ca	
114cb	
114cc	2
114cd	2
114ce	2
114cf	2



114d0	30
114d1	30
114d2	30
114d3	30
114d4	30
114d5	30
114d6	30
114d7	30
114d8	30
114d9	30
114da	30
114db	30
114dc	30
114dd	30
114de	30
114df	30
114e0	30
114e1	30
114e2	30
114e3	30
114e4	30
114e5	30
114e6	30
114e7	30
114e8	30
114e9	30
114ea	30
114eb	30
114ec	30
114ed	30
114ee	30
114ef	30
114f0	30
114f1	30
114f2	30
114f3	30
114f4	30
114f5	30
114f6	30
114f7	30
114f8	30
114f9	30
114fa	30
114fb	30
114fc	1900
114fd	1900
114fe	1900
114ff	1900
11500	1900
11501	1900
11502	1900
11503	1900
11504	1900
11505	1900
11506	1900
11507	1900
11508	1900
11509	1900
1150a	1900
1150b	1900
1150c	1900
1150d	1900
1150e	1900
1150f	1900
11510	1900
11511	1900
11512	1900
11513	1900
11514	1900



11515	1900
11516	1900
11517	1900
11518	1900
11519	1900
1151a	1900
1151b	1900
1151c	1900
1151d	1900
1151e	1900
1151f	1900
11520	1900
11521	1900
11522	1900
11523	1900
11524	1900
11525	1900
11526	1900
11527	1900
11528	1899
11529	1899
1152a	1899
1152b	1899
1152c	1929
1152d	1929
1152e	1929
1152f	1929
11530	1929
11531	1929
11532	1929
11533	1929
11534	29
11535	29
11536	29
11537	29
11538	29
11539	29
1153a	29
1153b	29
1153c	29
1153d	29
1153e	29
1153f	29
11540	29
11541	29
11542	29
11543	29
11544	29
11545	29
11546	29
11547	29
11548	29
11549	29
1154a	29
1154b	29
1154c	29
1154d	29
1154e	29
1154f	29
11550	29
11551	29
11552	29
11553	29
11554	29
11555	29
11556	29
11557	29
11558	29
11559	29



1155a	29
1155b	29
1155c	29
1155d	29
1155e	29
1155f	29
11560	29
11561	29
11562	29
11563	29
11564	29
11565	29
11566	29
11567	29
11568	29
11569	29
1156a	29
1156b	29
1156c	29
1156d	29
1156e	29
1156f	29
11570	29
11571	29
11572	29
11573	29
11574	

## 3.5 Instruction Timing Test

### Simulator

1. Edit Auto.cpp in the Axiom project so that the instruction-timing test is compiled.
2. Execute the simulator and wait until the test is finished.
3. Verify by looking in the message window that the total execution time is 36.0 +- 2 sec.

### Axiom Board

1. Set up the Axiom board per section 2.11.
2. At the monitor prompt, type mmw 2fc104 and hit return.
3. Type 00000002 return
4. Get a stop watch ready.
5. Type x and start the watch when you hit return.
6. Stop the watch at the prompt.
7. Verify 36 +- 2 seconds.